Enquêter dans les Graphes

Nathann Cohen[1] Nícolas A. Martins[2] **Fionn Mc Inerney**[3]
Nicolas Nisse[3] Stéphane Pérennes[3] Rudini Sampaio[2]

[1]CNRS, Univ Paris Sud, LRI, Orsay, France

[2]Universidade Federal do Ceará, Fortaleza, Brésil

[3]Université Côte d'Azur, Inria, CNRS, I3S, France

June 2, 2017
Time: Too early!
AlgoTel 2017

2-Player Combinatorial Games

- Mobile agents in a graph.
- Turn-by-turn with 2 players.
  - Coordination for common goal, e.g.,
    - Cops and Robbers (capture) (Quilliot, 1978 ; Nowakowski, Winkler, 1983 ; Bonato, Nowakowski, 2011)
    - Eternal Domination (protection) (Goddard et al, 2005 ; Klostermeyer, MacGillivray, 2009).

Investigations in Graphs

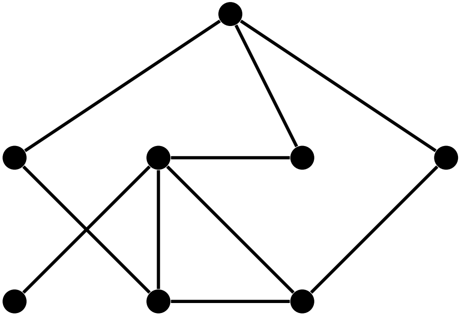
Suspect (1^{st}) vs detectives (2^{nd}) in a graph G.

Start : Suspect placed at a vertex. Then, detectives placed.

Turn-by-turn : Suspect traverses up to $\mathbf{s} \geq 2$ edges. Detectives traverse up to 1 edge.

Goal : Suspect wants to be at least distance $\mathbf{d}+1$ from all detectives.

Ex : $\mathbf{s} = 2$ and $\mathbf{d} = 1$.



Investigations in Graphs

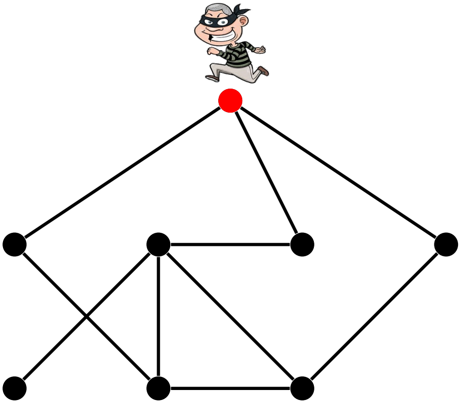
Suspect (1^{st}) vs detectives (2^{nd}) in a graph G.

Start : Suspect placed at a vertex. Then, detectives placed.

Turn-by-turn : Suspect traverses up to $\mathbf{s} \geq 2$ edges. Detectives traverse up to 1 edge.

Goal : Suspect wants to be at least distance $\mathbf{d}+1$ from all detectives.

Ex : $\mathbf{s} = 2$ and $\mathbf{d} = 1$.



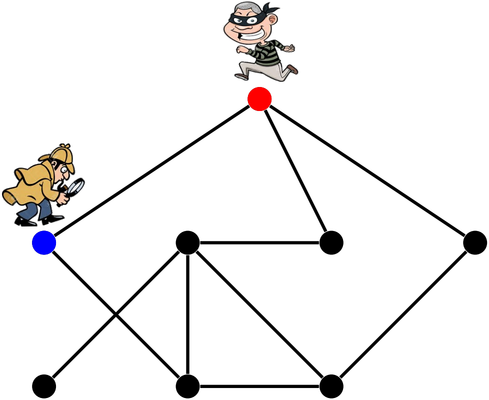
Suspect (1^{st}) vs detectives (2^{nd}) in a graph G.

Start : Suspect placed at a vertex. Then, detectives placed.

Turn-by-turn : Suspect traverses up to $\mathbf{s} \geq 2$ edges. Detectives traverse up to 1 edge.

Goal : Suspect wants to be at least distance $\mathbf{d}+1$ from all detectives.

Ex : $\mathbf{s} = 2$ and $\mathbf{d} = 1$.



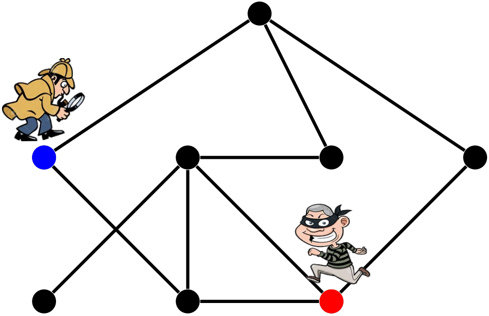
Suspect (1^{st}) vs detectives (2^{nd}) in a graph G.

Start : Suspect placed at a vertex. Then, detectives placed.

Turn-by-turn : Suspect traverses up to $\mathbf{s} \geq 2$ edges. Detectives traverse up to 1 edge.

Goal : Suspect wants to be at least distance $\mathbf{d}+1$ from all detectives.

Ex : $\mathbf{s} = 2$ and $\mathbf{d} = 1$.



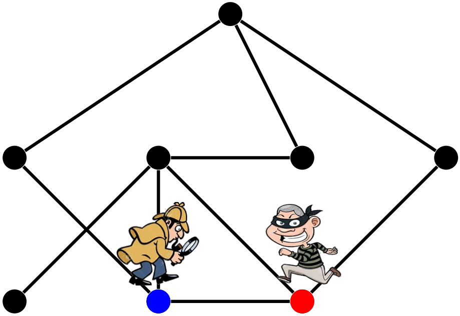
Suspect (1^{st}) vs detectives (2^{nd}) in a graph G.

Start : Suspect placed at a vertex. Then, detectives placed.

Turn-by-turn : Suspect traverses up to $\mathbf{s} \geq 2$ edges. Detectives traverse up to 1 edge.

Goal : Suspect wants to be at least distance $\mathbf{d}+1$ from all detectives.

Ex : $\mathbf{s} = 2$ and $\mathbf{d} = 1$.



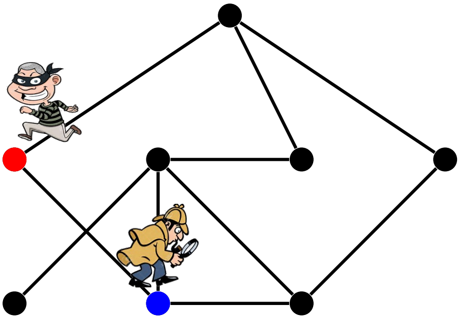
Suspect (1^{st}) vs detectives (2^{nd}) in a graph G.

Start : Suspect placed at a vertex. Then, detectives placed.

Turn-by-turn : Suspect traverses up to $\mathbf{s} \geq 2$ edges. Detectives traverse up to 1 edge.

Goal : Suspect wants to be at least distance $\mathbf{d} + 1$ from all detectives.

Ex : $\mathbf{s} = 2$ and $\mathbf{d} = 1$.



Investigations in Graphs

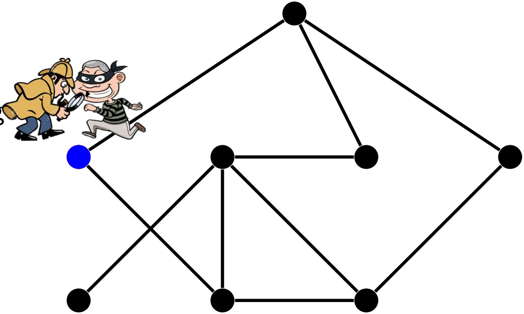
Suspect (1^{st}) vs detectives (2^{nd}) in a graph G.

Ex : $\mathbf{s} = 2$ and $\mathbf{d} = 1$.

Start : Suspect placed at a vertex. Then, detectives placed.

Turn-by-turn : Suspect traverses up to $\mathbf{s} \geq 2$ edges. Detectives traverse up to 1 edg

Goal : Suspect wants to be at least distance $\mathbf{d} + 1$ from all detectives.



Investigations in Graphs

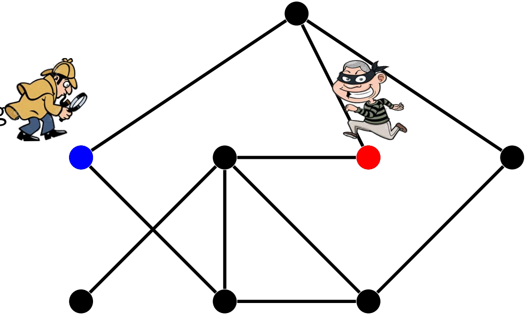
Suspect (1^{st}) vs detectives (2^{nd}) in a graph G.

Ex : $\mathbf{s} = 2$ and $\mathbf{d} = 1$.

Start : Suspect placed at a vertex. Then, detectives placed.

Turn-by-turn : Suspect traverses up to $\mathbf{s} \geq 2$ edges. Detectives traverse up to 1 edg

Goal : Suspect wants to be at least distance $\mathbf{d} + 1$ from all detectives.



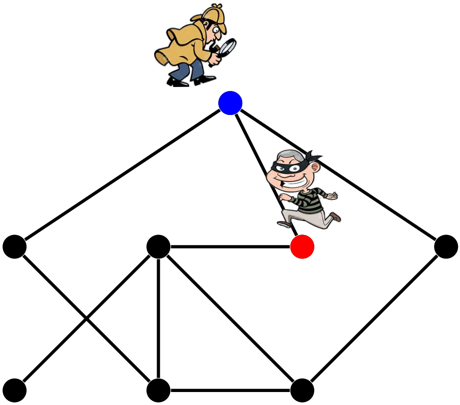
Suspect (1^{st}) vs detectives (2^{nd}) in a graph G.

Start : Suspect placed at a vertex. Then, detectives placed.

Turn-by-turn : Suspect traverses up to $\mathbf{s} \geq 2$ edges. Detectives traverse up to 1 edge.

Goal : Suspect wants to be at least distance $\mathbf{d}+1$ from all detectives.

Ex : $\mathbf{s} = 2$ and $\mathbf{d} = 1$.



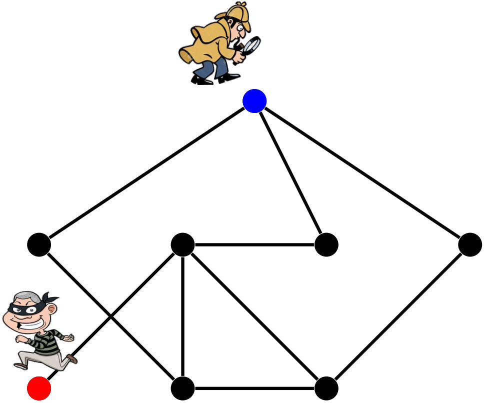
Suspect (1^{st}) vs detectives (2^{nd}) in a graph G.

Start : Suspect placed at a vertex. Then, detectives placed.

Turn-by-turn : Suspect traverses up to $\mathbf{s} \geq 2$ edges. Detectives traverse up to 1 edge.

Goal : Suspect wants to be at least distance $\mathbf{d} + 1$ from all detectives.

Ex : $\mathbf{s} = 2$ and $\mathbf{d} = 1$.



Guard Number : $gn_{s,d}(G)$

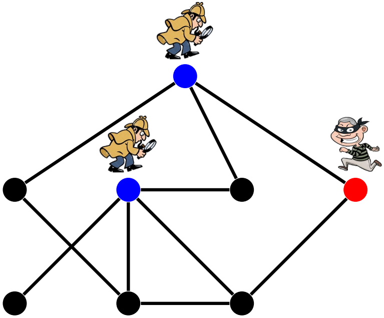
Definition

For all $s \geq 2$, $d \geq 0$ and a graph G, $gn_{s,d}(G)$ is the minimum number of detectives guaranteed to win vs the suspect.

Guard Number : $gn_{s,d}(G)$

Definition

For all $s \geq 2$, $d \geq 0$ and a graph G, $gn_{s,d}(G)$ is the minimum number of detectives guaranteed to win vs the suspect.



$$gn_{2,1}(G) = 2$$

$$gn_{s,1}(G) \leq \gamma(G)$$

Our Results : Computing *gn*

Complexity

Calculating $gn_{s,d}$ is NP-hard in general.

Tight bounds for paths

$gn_{s,d}(P_n) = \left\lceil \frac{n}{2d+2+q} \right\rceil$ where $q = \lfloor \frac{2d}{s-1} \rfloor$.

Almost tight bounds for cycles

$gn_{s,d}(C_n) \backsimeq \left\lceil \frac{n+2q}{2(d+q)+3} \right\rceil$ where $q = \lfloor \frac{2d}{s-1} \rfloor$.

Polynomial time Linear Program for trees

Can calculate $gn_{s,d}(T)$ and a corresp. strategy in polynomial time.

Grids

$\exists \beta > 0$, s.t. $\Omega(n^{1+\beta}) \leq gn_{s,d}(G_{n \times n})$.

Related Work

- Cops vs robber (capture at a distance) (Bonato et al, 2010).

Related Work

- Cops vs robber (capture at a distance) (Bonato et al, 2010).
- Cops vs fast robber (Fomin et al, 2010).

Related Work

- Cops vs robber (capture at a distance) (Bonato et al, 2010).
- Cops vs fast robber (Fomin et al, 2010).
    - Need $e^{\Omega(log n / log log n)}$ cops in $n \times n$ grid. (Balister et al, 2016).

Related Work

- Cops vs robber (capture at a distance) (Bonato et al, 2010).
- Cops vs fast robber (Fomin et al, 2010).
  - Need $e^{\Omega(logn/loglogn)}$ cops in $n \times n$ grid. (Balister et al, 2016).
- Eternal Domination (Goddard et al, 2005).

Related Work

- Cops vs robber (capture at a distance) (Bonato et al, 2010).
- Cops vs fast robber (Fomin et al, 2010).
  - Need $e^{\Omega(logn/loglogn)}$ cops in $n \times n$ grid. (Balister et al, 2016).
- Eternal Domination (Goddard et al, 2005).
  - $\lceil \frac{4n}{5} \rceil + 1 \leq \gamma^m(3 \times n\ grid) \leq \lceil \frac{4n}{5} \rceil + 3$ (Delaney et al, 2015).

Related Work

- Cops vs robber (capture at a distance) (Bonato et al, 2010).
- Cops vs fast robber (Fomin et al, 2010).
  - Need $e^{\Omega(logn/loglogn)}$ cops in $n \times n$ grid. (Balister et al, 2016).
- Eternal Domination (Goddard et al, 2005).
  - $\lceil \frac{4n}{5} \rceil + 1 \leq \gamma^m(3 \times n\ grid) \leq \lceil \frac{4n}{5} \rceil + 3$ (Delaney et al, 2015).
  - $\gamma^m(G) = gn_{s,d}(G)$ when $s = \infty$ and $d = 0$.

Paths

Theorem

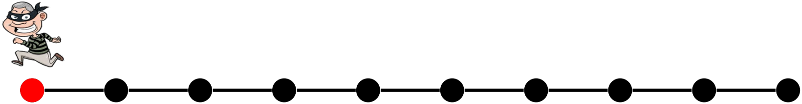
For all $s \geq 2$, $d \geq 0$, and a path P_n on n vertices,

$$gn_{s,d}(P_n) = \left\lceil \frac{n}{2d+2+\lfloor \frac{2d}{s-1} \rfloor} \right\rceil$$

Theorem

For all $s \geq 2$, $d \geq 0$, and a path P_n on n vertices,
$$gn_{s,d}(P_n) = \left\lceil \frac{n}{2d+2+\lfloor \frac{2d}{s-1} \rfloor} \right\rceil$$

Ex : $s = 3$ and $d = 1$.



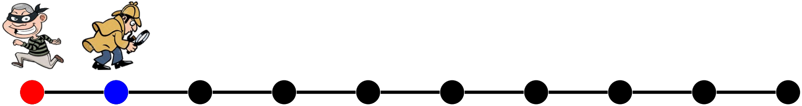
Paths : Lower bound

Theorem

For all $s \geq 2$, $d \geq 0$, and a path P_n on n vertices,

$$gn_{s,d}(P_n) = \left\lceil \frac{n}{2d+2+\lfloor \frac{2d}{s-1} \rfloor} \right\rceil$$

Ex : $s = 3$ and $d = 1$.



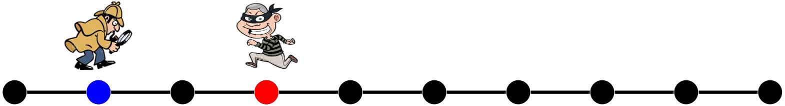
Paths : Lower bound

Theorem

For all $s \geq 2$, $d \geq 0$, and a path P_n on n vertices,

$$gn_{s,d}(P_n) = \left\lceil \frac{n}{2d+2+\lfloor \frac{2d}{s-1} \rfloor} \right\rceil$$

Ex : $s = 3$ and $d = 1$.

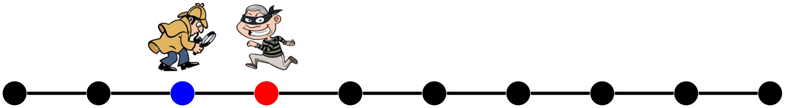


Theorem

For all $s \geq 2$, $d \geq 0$, and a path P_n on n vertices,

$$gn_{s,d}(P_n) = \left\lceil \frac{n}{2d+2+\lfloor \frac{2d}{s-1} \rfloor} \right\rceil$$

Ex : $s = 3$ and $d = 1$.



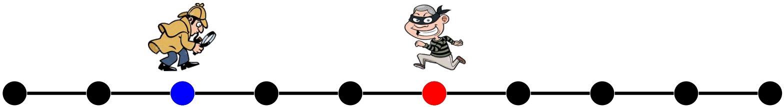
Paths : Lower bound

Theorem

For all $s \geq 2$, $d \geq 0$, and a path P_n on n vertices,

$$gn_{s,d}(P_n) = \left\lceil \frac{n}{2d+2+\lfloor \frac{2d}{s-1} \rfloor} \right\rceil$$

Ex : $s = 3$ and $d = 1$.

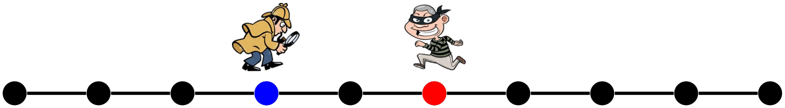


Paths : Lower bound

Theorem

For all $s \geq 2$, $d \geq 0$, and a path P_n on n vertices,
$gn_{s,d}(P_n) = \left\lceil \frac{n}{2d+2+\lfloor \frac{2d}{s-1} \rfloor} \right\rceil$

Ex : $s = 3$ and $d = 1$.



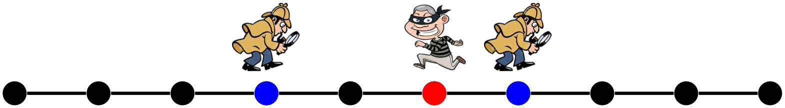
Paths : Lower bound

Theorem

For all $s \geq 2$, $d \geq 0$, and a path P_n on n vertices,

$$gn_{s,d}(P_n) = \left\lceil \frac{n}{2d+2+\lfloor \frac{2d}{s-1} \rfloor} \right\rceil$$

Ex : $s = 3$ and $d = 1$.



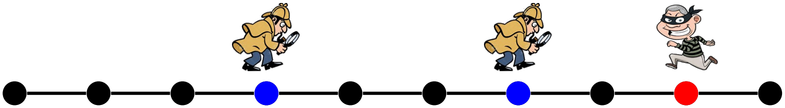
Paths : Lower bound

Theorem

For all $s \geq 2$, $d \geq 0$, and a path P_n on n vertices,

$$gn_{s,d}(P_n) = \left\lceil \frac{n}{2d+2+\lfloor \frac{2d}{s-1} \rfloor} \right\rceil$$

Ex : $s = 3$ and $d = 1$.



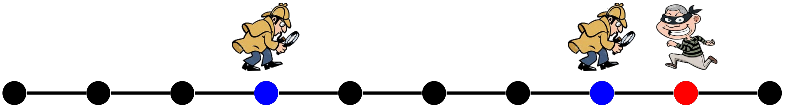
Paths : Lower bound

Theorem

For all $s \geq 2$, $d \geq 0$, and a path P_n on n vertices,

$$gn_{s,d}(P_n) = \left\lceil \frac{n}{2d+2+\lfloor \frac{2d}{s-1} \rfloor} \right\rceil$$

Ex : $s = 3$ and $d = 1$.

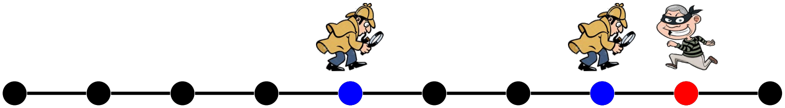


Paths : Lower bound

Theorem

For all $s \geq 2$, $d \geq 0$, and a path P_n on n vertices,
$gn_{s,d}(P_n) = \left\lceil \frac{n}{2d+2+\lfloor \frac{2d}{s-1} \rfloor} \right\rceil$

Ex : $s = 3$ and $d = 1$.



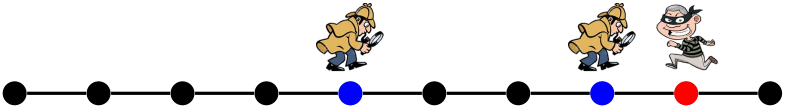
Paths : Lower bound

Theorem

For all $s \geq 2$, $d \geq 0$, and a path P_n on n vertices,

$$gn_{s,d}(P_n) = \left\lceil \frac{n}{2d+2+\lfloor \frac{2d}{s-1} \rfloor} \right\rceil$$

Ex : $s = 3$ and $d = 1$.



$$gn_{3,1}(P_{10}) = 2$$

Paths : Upper bound

Theorem

For all $s \geq 2$, $d \geq 0$, and a path P_n on n vertices,

$$gn_{s,d}(P_n) = \left\lceil \frac{n}{2d+2+\lfloor \frac{2d}{s-1} \rfloor} \right\rceil$$

Ex : $s = 3$ and $d = 1$.

1 detective can protect subpath of $2d + 2 + \lfloor \frac{2d}{s-1} \rfloor$ vertices.

Paths : Upper bound

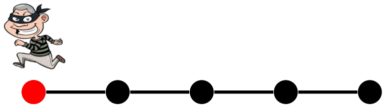
Theorem

For all $s \geq 2$, $d \geq 0$, and a path P_n on n vertices,

$$gn_{s,d}(P_n) = \left\lceil \frac{n}{2d+2+\lfloor \frac{2d}{s-1} \rfloor} \right\rceil$$

Ex : $s = 3$ and $d = 1$.

1 detective can protect subpath of $2d + 2 + \lfloor \frac{2d}{s-1} \rfloor$ vertices.



Paths : Upper bound

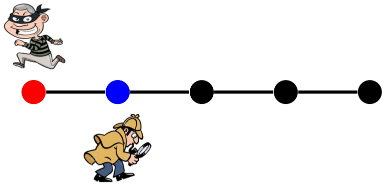
Theorem

For all $s \geq 2$, $d \geq 0$, and a path P_n on n vertices,

$$gn_{s,d}(P_n) = \left\lceil \frac{n}{2d+2+\lfloor \frac{2d}{s-1} \rfloor} \right\rceil$$

Ex : $s = 3$ and $d = 1$.

1 detective can protect subpath of $2d + 2 + \lfloor \frac{2d}{s-1} \rfloor$ vertices.



Paths : Upper bound

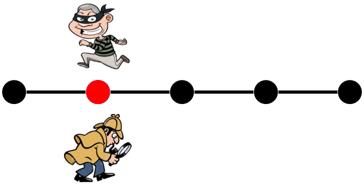
Theorem

For all $s \geq 2$, $d \geq 0$, and a path P_n on n vertices,

$$gn_{s,d}(P_n) = \left\lceil \frac{n}{2d+2+\lfloor \frac{2d}{s-1} \rfloor} \right\rceil$$

Ex : $s = 3$ and $d = 1$.

1 detective can protect subpath of $2d + 2 + \lfloor \frac{2d}{s-1} \rfloor$ vertices.



Paths : Upper bound

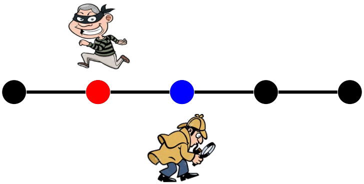
Theorem

For all $s \geq 2$, $d \geq 0$, and a path P_n on n vertices,

$$gn_{s,d}(P_n) = \left\lceil \frac{n}{2d+2+\lfloor \frac{2d}{s-1} \rfloor} \right\rceil$$

Ex : $s = 3$ and $d = 1$.

1 detective can protect subpath of $2d + 2 + \lfloor \frac{2d}{s-1} \rfloor$ vertices.



Paths : Upper bound

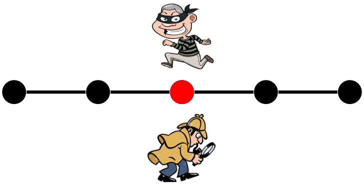
Theorem

For all $s \geq 2$, $d \geq 0$, and a path P_n on n vertices,

$$gn_{s,d}(P_n) = \left\lceil \frac{n}{2d+2+\lfloor \frac{2d}{s-1} \rfloor} \right\rceil$$

Ex : $s = 3$ and $d = 1$.

1 detective can protect subpath of $2d + 2 + \lfloor \frac{2d}{s-1} \rfloor$ vertices.



Paths : Upper bound

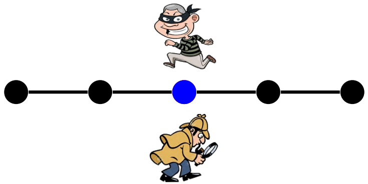
Theorem

For all $s \geq 2$, $d \geq 0$, and a path P_n on n vertices,

$$gn_{s,d}(P_n) = \left\lceil \frac{n}{2d+2+\lfloor \frac{2d}{s-1} \rfloor} \right\rceil$$

Ex : $s = 3$ and $d = 1$.

1 detective can protect subpath of $2d + 2 + \lfloor \frac{2d}{s-1} \rfloor$ vertices.



Paths : Upper bound

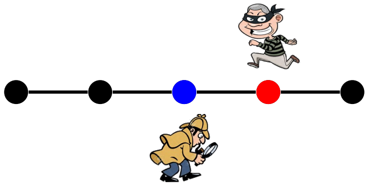
Theorem

For all $s \geq 2$, $d \geq 0$, and a path P_n on n vertices,

$$gn_{s,d}(P_n) = \left\lceil \frac{n}{2d+2+\lfloor \frac{2d}{s-1} \rfloor} \right\rceil$$

Ex : $s = 3$ and $d = 1$.

1 detective can protect subpath of $2d + 2 + \lfloor \frac{2d}{s-1} \rfloor$ vertices.



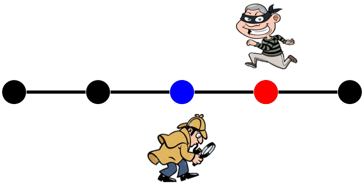
Paths : Upper bound

Theorem

For all $s \geq 2$, $d \geq 0$, and a path P_n on n vertices,
$gn_{s,d}(P_n) = \left\lceil \frac{n}{2d+2+\lfloor \frac{2d}{s-1} \rfloor} \right\rceil$

Ex : $s = 3$ and $d = 1$.

1 detective can protect subpath of $2d + 2 + \lfloor \frac{2d}{s-1} \rfloor$ vertices.



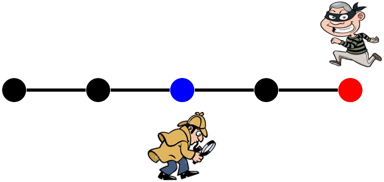
Paths : Upper bound

Theorem

For all $s \geq 2$, $d \geq 0$, and a path P_n on n vertices,
$gn_{s,d}(P_n) = \left\lceil \frac{n}{2d+2+\lfloor \frac{2d}{s-1} \rfloor} \right\rceil$

Ex : $s = 3$ and $d = 1$.

1 detective can protect subpath of $2d + 2 + \lfloor \frac{2d}{s-1} \rfloor$ vertices.



Paths : Upper bound

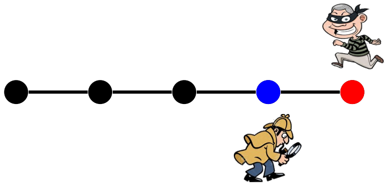
Theorem

For all $s \geq 2$, $d \geq 0$, and a path P_n on n vertices,

$gn_{s,d}(P_n) = \left\lceil \frac{n}{2d+2+\lfloor \frac{2d}{s-1} \rfloor} \right\rceil$

Ex : $s = 3$ and $d = 1$.

1 detective can protect subpath of $2d + 2 + \lfloor \frac{2d}{s-1} \rfloor$ vertices.



Paths : Upper bound

Theorem

For all $s \geq 2$, $d \geq 0$, and a path P_n on n vertices,

$$gn_{s,d}(P_n) = \left\lceil \frac{n}{2d+2+\lfloor \frac{2d}{s-1} \rfloor} \right\rceil$$

Each detective protects a subpath of $2d + 2 + \lfloor \frac{2d}{s-1} \rfloor$ vertices.

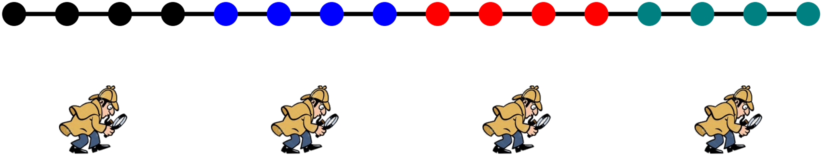
Paths : Upper bound

Theorem

For all $s \geq 2$, $d \geq 0$, and a path P_n on n vertices,

$$gn_{s,d}(P_n) = \left\lceil \frac{n}{2d+2+\lfloor \frac{2d}{s-1} \rfloor} \right\rceil$$

Each detective protects a subpath of $2d + 2 + \lfloor \frac{2d}{s-1} \rfloor$ vertices.



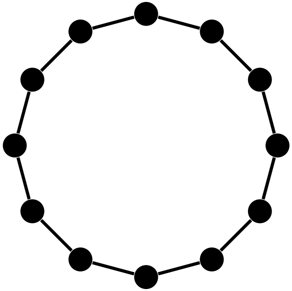
Cycles : Upper Bound Case $2d < s - 1$

Theorem

For all $s \geq 2$, $d \geq 0$ s.t. $2d < s - 1$, and a cycle C_n on n vertices, $gn_{s,d}(C_n) = \left\lceil \frac{n}{2d+3} \right\rceil$.

Ex : $s = 6$ and $d = 0$.

$$gn_{6,0}(C_{12}) = 4$$



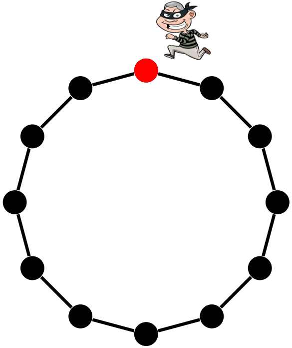
Cycles : Upper Bound Case $2d < s - 1$

Theorem

For all $s \geq 2$, $d \geq 0$ s.t. $2d < s - 1$, and a cycle C_n on n vertices, $gn_{s,d}(C_n) = \left\lceil \frac{n}{2d+3} \right\rceil$.

Ex : $s = 6$ and $d = 0$.

$$gn_{6,0}(C_{12}) = 4$$

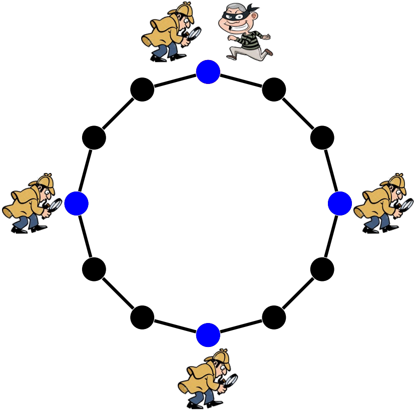


Theorem

For all $s \geq 2$, $d \geq 0$ s.t. $2d < s - 1$, and a cycle C_n on n vertices, $gn_{s,d}(C_n) = \left\lceil \frac{n}{2d+3} \right\rceil$.

Ex : $s = 6$ and $d = 0$.

$$gn_{6,0}(C_{12}) = 4$$



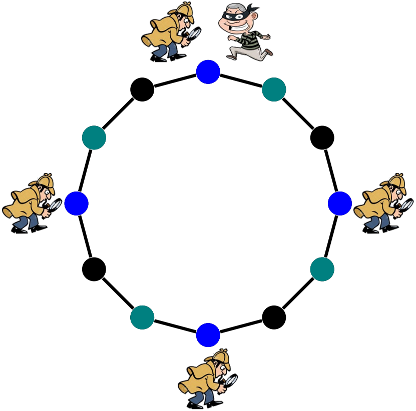
Cycles : Upper Bound Case $2d < s - 1$

Theorem

For all $s \geq 2$, $d \geq 0$ s.t. $2d < s-1$, and a cycle C_n on n vertices, $gn_{s,d}(C_n) = \left\lceil \frac{n}{2d+3} \right\rceil$.

Ex : $s = 6$ and $d = 0$.

$$gn_{6,0}(C_{12}) = 4$$



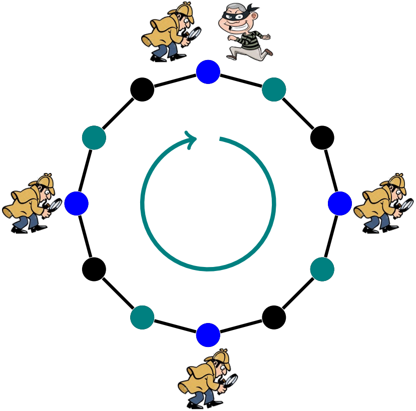
Cycles : Upper Bound Case $2d < s - 1$

Theorem

For all $s \geq 2$, $d \geq 0$ s.t. $2d < s - 1$, and a cycle C_n on n vertices,
$gn_{s,d}(C_n) = \left\lceil \frac{n}{2d+3} \right\rceil$.

Ex : $s = 6$ and $d = 0$.

$$gn_{6,0}(C_{12}) = 4$$



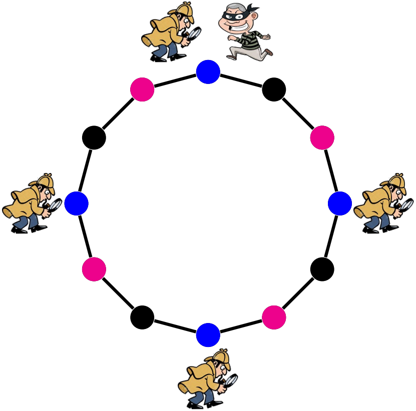
Cycles : Upper Bound Case $2d < s - 1$

Theorem

For all $s \geq 2$, $d \geq 0$ s.t. $2d < s - 1$, and a cycle C_n on n vertices, $gn_{s,d}(C_n) = \left\lceil \frac{n}{2d+3} \right\rceil$.

Ex : $s = 6$ and $d = 0$.

$$gn_{6,0}(C_{12}) = 4$$

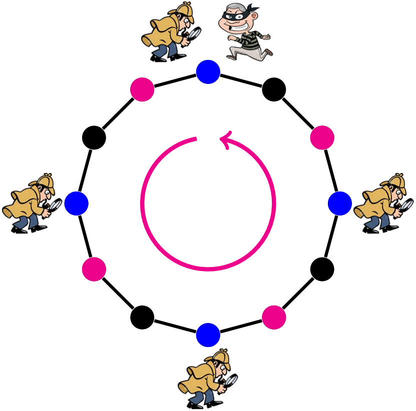


Theorem

For all $s \geq 2$, $d \geq 0$ s.t. $2d < s - 1$, and a cycle C_n on n vertices, $gn_{s,d}(C_n) = \left\lceil \frac{n}{2d+3} \right\rceil$.

Ex : $s = 6$ and $d = 0$.

$$gn_{6,0}(C_{12}) = 4$$



Cycles

Let $2d = q(s-1) + r$ $(0 \leq r < s-1)$ and $2d = q's + r'$ $(0 \leq r' < s)$.
Then, $q = \left\lfloor \frac{2d}{s-1} \right\rfloor$ and $q' = \left\lfloor \frac{2d}{s} \right\rfloor$.
Let $(q^*, r^*) = (q, r)$ if s is odd and $(q^*, r^*) = (q', r')$ otherwise.

Theorem

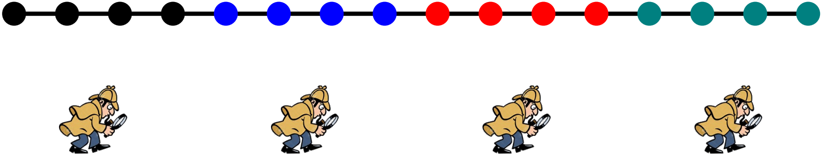
For all $s \geq 2$, $d \geq 0$ s.t. $q = 0$, and a cycle C_n on n vertices,
$gn_{s,d}(C_n) = \left\lceil \frac{n}{2d+3} \right\rceil$.

Theorem

For all $s \geq 2$, $d \geq 0$ s.t. $q \neq 0$, and a cycle C_n on n vertices,
$\left\lceil \frac{n+2q}{2(d+q)+3} \right\rceil \leq gn_{s,d}(C_n) \leq \left\lceil \frac{n+2q^*}{2(d+q^*)-r^*} \right\rceil$.

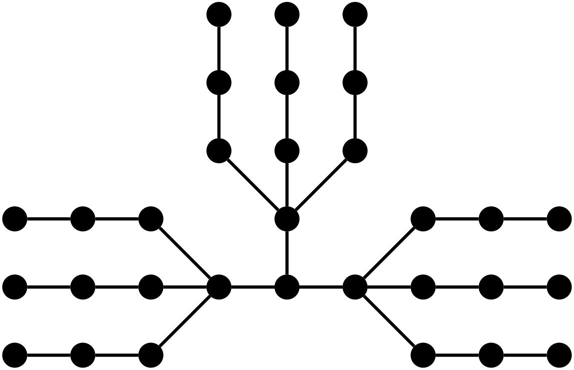
Trees are Harder

Paths : 1 detective per subpath of $2d + 2 + \lfloor \frac{2d}{s-1} \rfloor$ vertices.



Trees are Harder

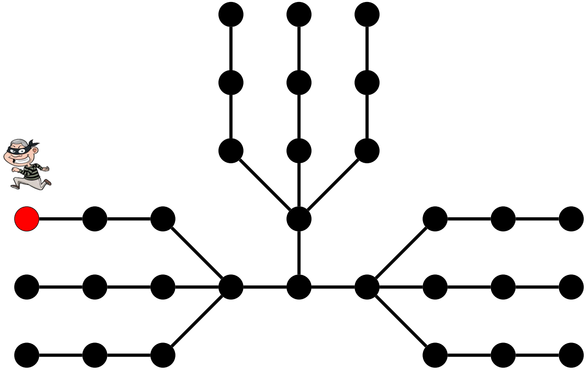
Can't always divide tree into subtrees protected by a certain number of detectives.



Example of a tree T where $s = 2$, $d = 1$ and $gn_{2,1}(T) = 4$.

Trees are Harder

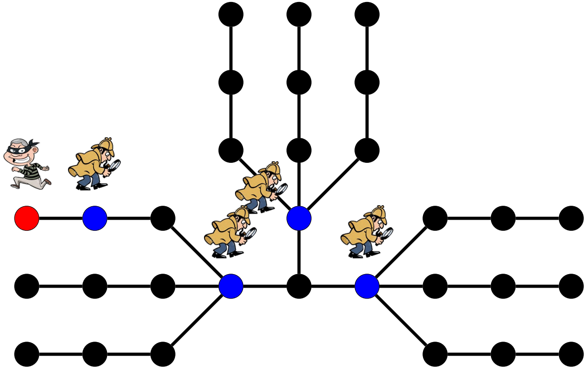
Can't always divide tree into subtrees protected by a certain number of detectives.



Example of a tree T where $s = 2$, $d = 1$ and $gn_{2,1}(T) = 4$.

Trees are Harder

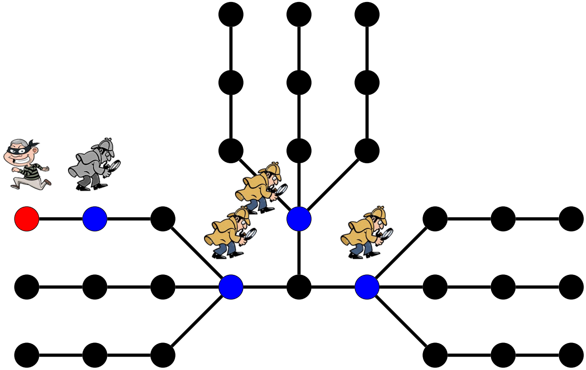
Can't always divide tree into subtrees protected by a certain number of detectives.



Example of a tree T where $s = 2$, $d = 1$ and $gn_{2,1}(T) = 4$.

Trees are Harder

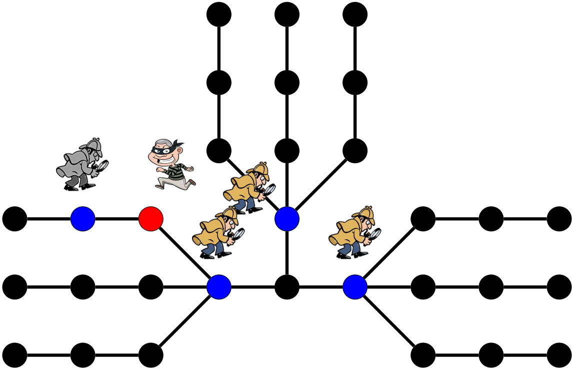
Can't always divide tree into subtrees protected by a certain number of detectives.



Example of a tree T where $s = 2$, $d = 1$ and $gn_{2,1}(T) = 4$.

Trees are Harder

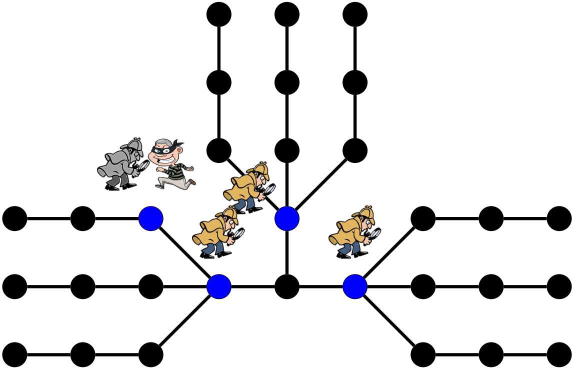
Can't always divide tree into subtrees protected by a certain number of detectives.



Example of a tree T where $s = 2$, $d = 1$ and $gn_{2,1}(T) = 4$.

Trees are Harder

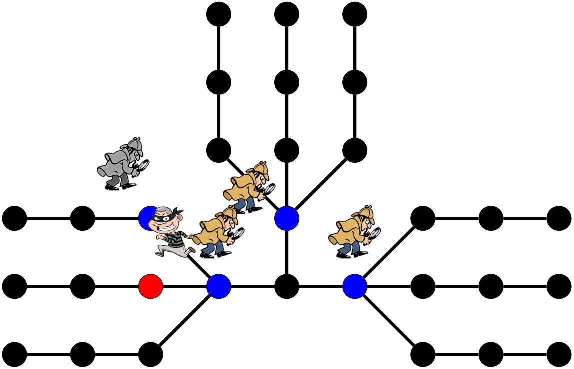
Can't always divide tree into subtrees protected by a certain number of detectives.



Example of a tree T where $s = 2$, $d = 1$ and $gn_{2,1}(T) = 4$.

Trees are Harder

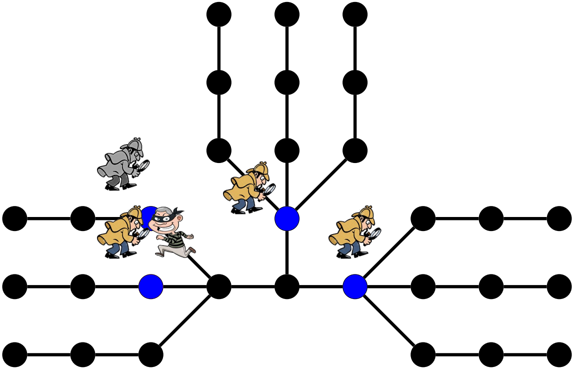
Can't always divide tree into subtrees protected by a certain number of detectives.



Example of a tree T where $s = 2$, $d = 1$ and $gn_{2,1}(T) = 4$.

Trees are Harder

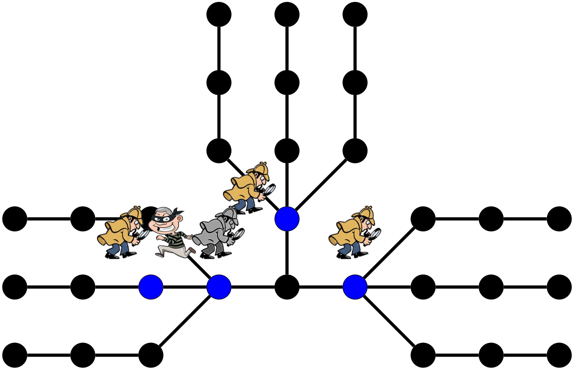
Can't always divide tree into subtrees protected by a certain number of detectives.



Example of a tree T where $s = 2$, $d = 1$ and $gn_{2,1}(T) = 4$.

Trees are Harder

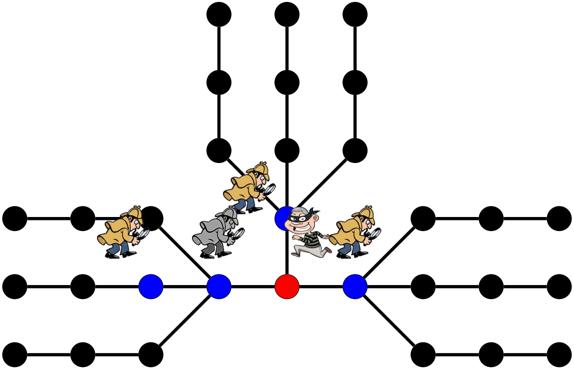
Can't always divide tree into subtrees protected by a certain number of detectives.



Example of a tree T where $s = 2$, $d = 1$ and $gn_{2,1}(T) = 4$.

Trees are Harder

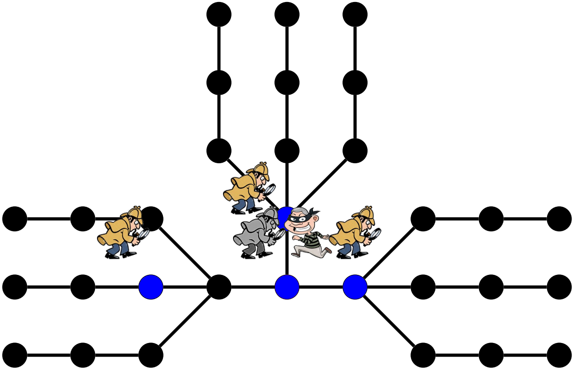
Can't always divide tree into subtrees protected by a certain number of detectives.



Example of a tree T where $s = 2$, $d = 1$ and $gn_{2,1}(T) = 4$.

Trees are Harder

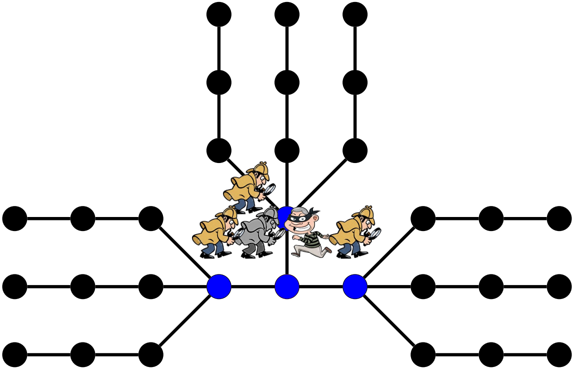
Can't always divide tree into subtrees protected by a certain number of detectives.



Example of a tree T where $s = 2$, $d = 1$ and $gn_{2,1}(T) = 4$.

Trees are Harder

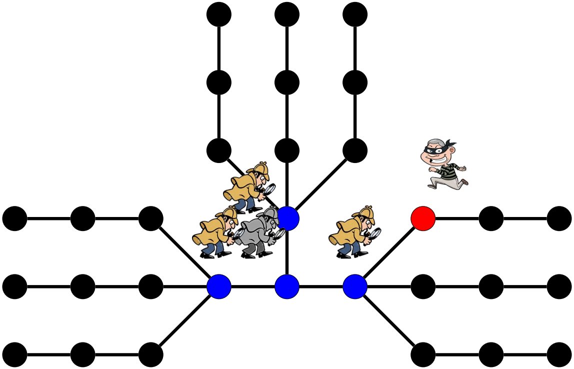
Can't always divide tree into subtrees protected by a certain number of detectives.



Example of a tree T where $s = 2$, $d = 1$ and $gn_{2,1}(T) = 4$.

Trees are Harder

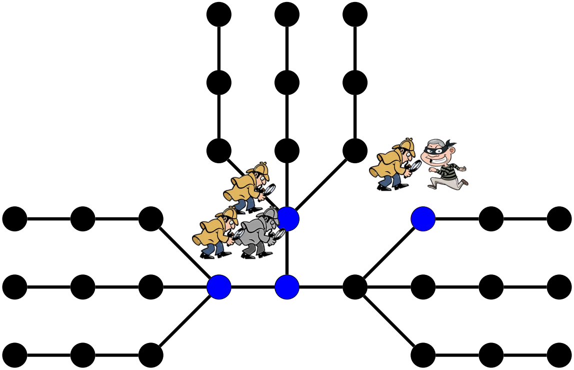
Can't always divide tree into subtrees protected by a certain number of detectives.



Example of a tree T where $s = 2$, $d = 1$ and $gn_{2,1}(T) = 4$.

Trees are Harder

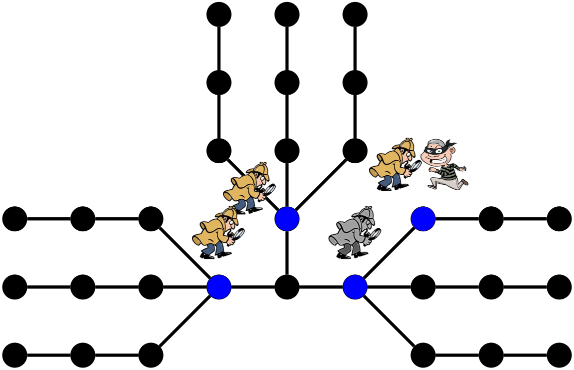
Can't always divide tree into subtrees protected by a certain number of detectives.



Example of a tree T where $s = 2$, $d = 1$ and $gn_{2,1}(T) = 4$.

Trees are Harder

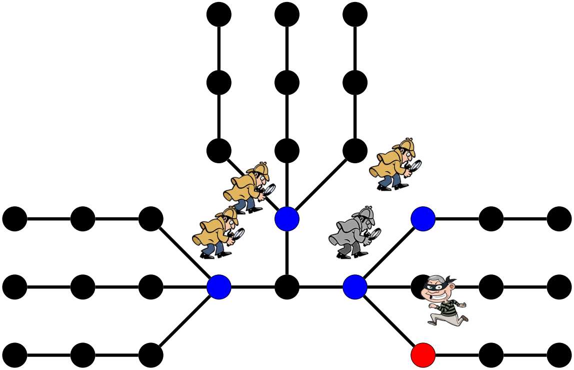
Can't always divide tree into subtrees protected by a certain number of detectives.



Example of a tree T where $s = 2$, $d = 1$ and $gn_{2,1}(T) = 4$.

Trees are Harder

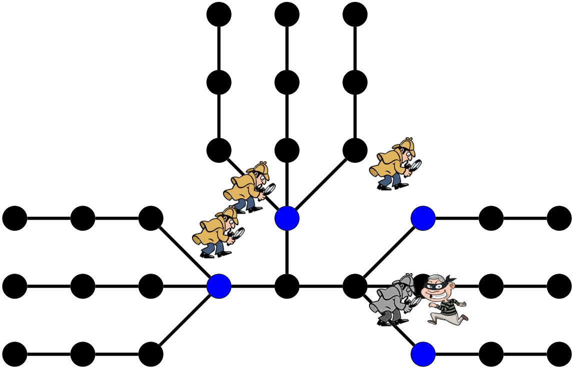
Can't always divide tree into subtrees protected by a certain number of detectives.



Example of a tree T where $s = 2$, $d = 1$ and $gn_{2,1}(T) = 4$.

Trees are Harder

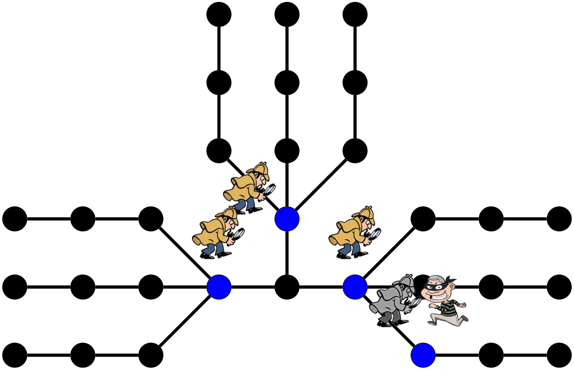
Can't always divide tree into subtrees protected by a certain number of detectives.



Example of a tree T where $s = 2$, $d = 1$ and $gn_{2,1}(T) = 4$.

Trees are Harder

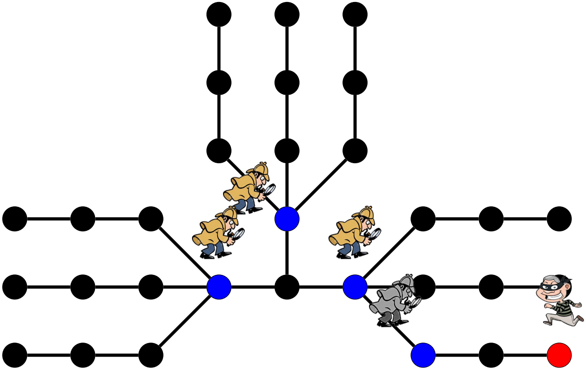
Can't always divide tree into subtrees protected by a certain number of detectives.



Example of a tree T where $s = 2$, $d = 1$ and $gn_{2,1}(T) = 4$.

Trees are Harder

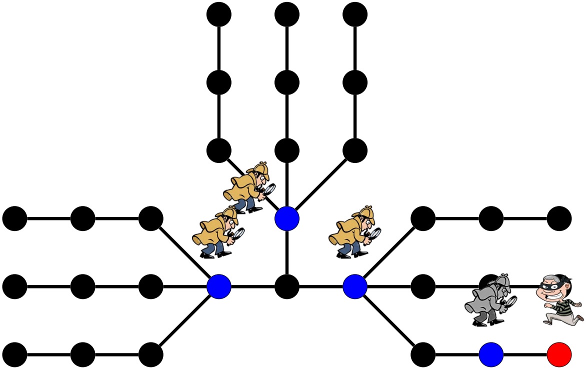
Can't always divide tree into subtrees protected by a certain number of detectives.



Example of a tree T where $s = 2$, $d = 1$ and $gn_{2,1}(T) = 4$.

Trees are Harder

Can't always divide tree into subtrees protected by a certain number of detectives.



Example of a tree T where $s = 2$, $d = 1$ and $gn_{2,1}(T) = 4$.

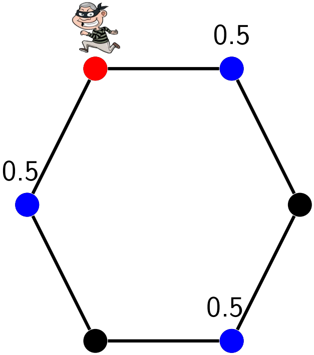
Fractional Version of the Game

- Detectives may be fractional entities ; movements rep. by flows.
- Unchanged for suspect. Total fraction of detectives distance $\leq d$ from suspect must be ≥ 1.

Fractional Version of the Game

- Detectives may be fractional entities ; movements rep. by flows.
- Unchanged for suspect. Total fraction of detectives distance $\leq d$ from suspect must be ≥ 1.

$s = 2,\ d = 1.$



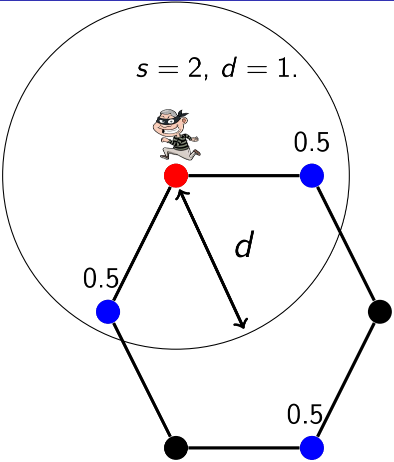
$gn_{2,1}(C_6) = 2$

but

1.5 detectives suffice.

Fractional Version of the Game

- Detectives may be fractional entities ; movements rep. by flows.
- Unchanged for suspect. Total fraction of detectives distance $\leq d$ from suspect must be ≥ 1.

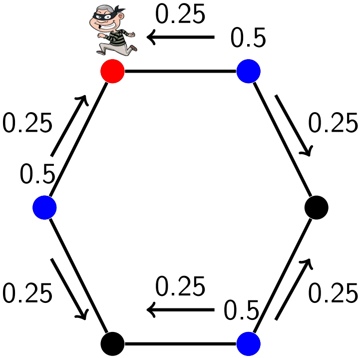


$gn_{2,1}(C_6) = 2$
but
1.5 detectives suffice.

Fractional Version of the Game

- Detectives may be fractional entities ; movements rep. by flows.
- Unchanged for suspect. Total fraction of detectives distance $\leq d$ from suspect must be ≥ 1.

$s = 2,\ d = 1.$

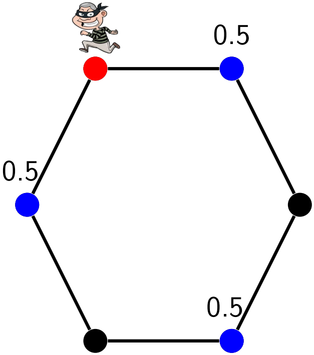


$gn_{2,1}(C_6) = 2$
but
1.5 detectives suffice.

Fractional Version of the Game

- Detectives may be fractional entities ; movements rep. by flows.
- Unchanged for suspect. Total fraction of detectives distance $\leq d$ from suspect must be ≥ 1.
- Linear program to compute optimal fractional strategy.

$s = 2,\ d = 1.$



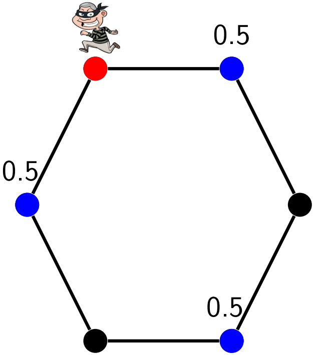
$gn_{2,1}(C_6) = 2$
but
1.5 detectives suffice.

Fractional Version of the Game

- Detectives may be fractional entities ; movements rep. by flows.
- Unchanged for suspect. Total fraction of detectives distance $\leq d$ from suspect must be ≥ 1.

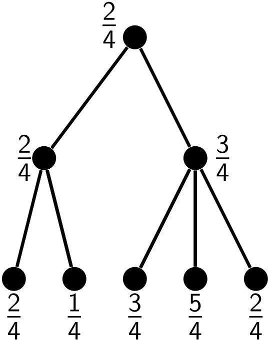
- Linear program to compute optimal fractional strategy.
- Optimal fractional strategy $\Rightarrow$ optimal integral strategy in trees.

$s = 2,\ d = 1.$



$gn_{2,1}(C_6) = 2$
but
1.5 detectives suffice.

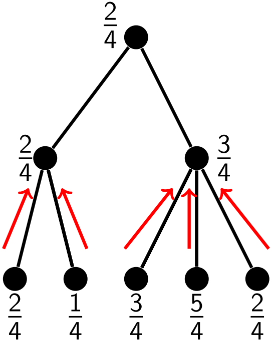
Theorem : Can transform optimal fractional strategy into optimal integral strategy in polynomial time.



Fractional Conf.

Opt. Fractional Strategy ⇒ Opt. Integral Strategy in Trees

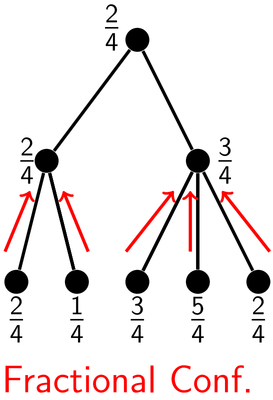
Theorem : Can transform optimal fractional strategy into optimal integral strategy in polynomial time.

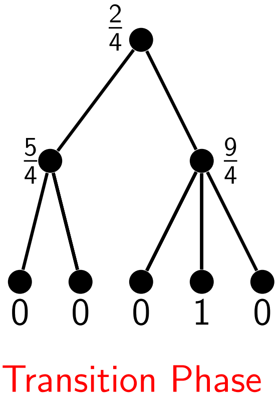


Fractional Conf.

Opt. Fractional Strategy ⇒ Opt. Integral Strategy in Trees

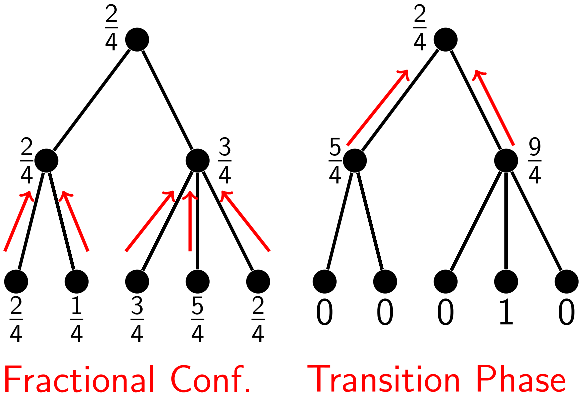
Theorem : Can transform optimal fractional strategy into optimal integral strategy in polynomial time.





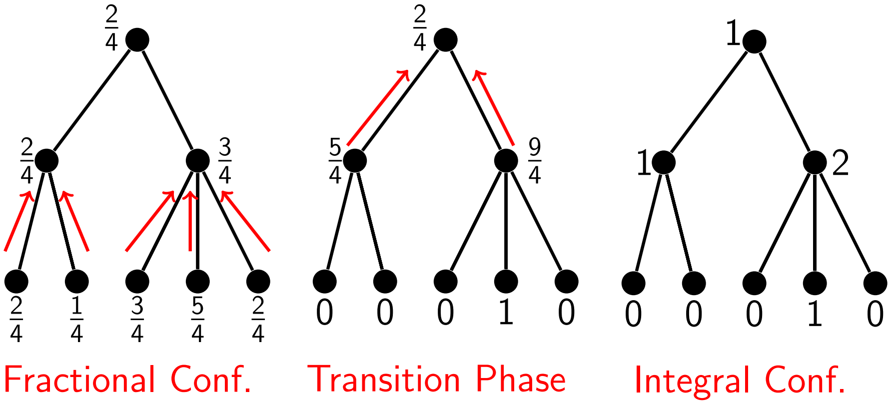
Opt. Fractional Strategy ⇒ Opt. Integral Strategy in Trees

Theorem : Can transform optimal fractional strategy into optimal integral strategy in polynomial time.



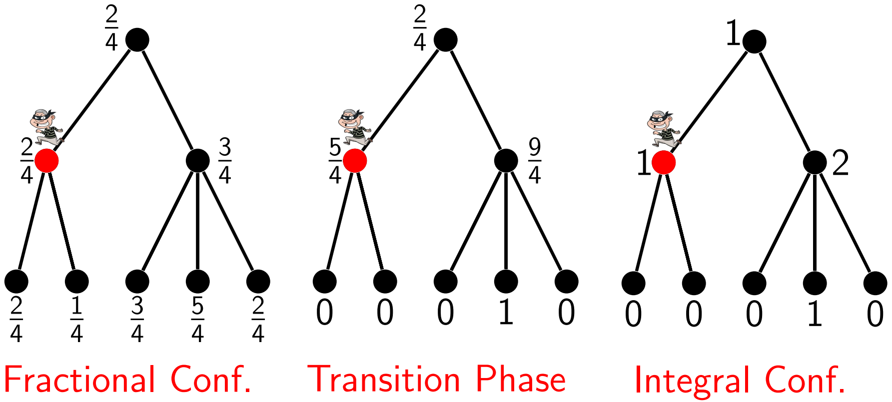
Opt. Fractional Strategy ⇒ Opt. Integral Strategy in Trees

Theorem : Can transform optimal fractional strategy into optimal integral strategy in polynomial time.



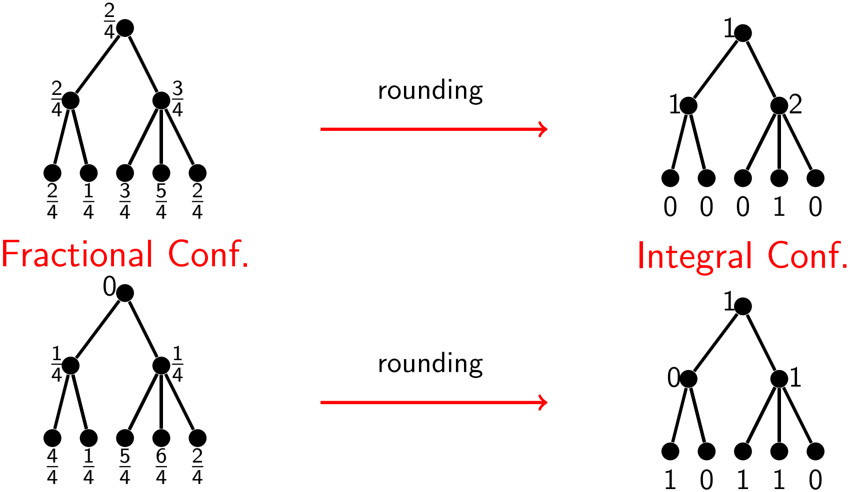
Opt. Fractional Strategy ⇒ Opt. Integral Strategy in Trees

Theorem : Can transform optimal fractional strategy into optimal integral strategy in polynomial time.



Tree's protection and detectives' movements preserved.

Opt. Fractional Strategy ⇒ Opt. Integral Strategy in Trees



Tree's protection and detectives' movements preserved.

Restricted Strategies

$f : V^k \times V \Rightarrow V^k$ (Unrestricted strategy)

$\omega : V \Rightarrow V^k$ (Restricted strategy)

Restricted Strategies

$f : V^k \times V \Rightarrow V^k$ (Unrestricted strategy)

$\omega : V \Rightarrow V^k$ (Restricted strategy)

- Detectives' positions depend only on position of suspect.
- 1 Unique configuration for detectives for each position of suspect.

Restricted Strategies

$f : V^k \times V \Rightarrow V^k$ (Unrestricted strategy)

$\omega : V \Rightarrow V^k$ (Restricted strategy)

- Detectives' positions depend only on position of suspect.
- 1 Unique configuration for detectives for each position of suspect.

Theorem

Optimal fractional strategy $\Rightarrow$ optimal fractional restricted strategy in trees.

Restricted Strategies

$f : V^k \times V \Rightarrow V^k$ (Unrestricted strategy)

$\omega : V \Rightarrow V^k$ (Restricted strategy)

- Detectives' positions depend only on position of suspect.
- 1 Unique configuration for detectives for each position of suspect.

Theorem

Optimal fractional strategy $\Rightarrow$ optimal fractional restricted strategy in trees.

Can calculate optimal restricted fractional strategies with Linear Program in polynomial time.

Linear Program to Compute Restricted Strategy

Restricted strategy : $\omega : V \Rightarrow V^k$

$\omega_{x,u}$: quantity of detectives on u when suspect is on x.

$f_{x,x',u,u'}$: quantity of detectives that go from u to u' when suspect goes from x to x'.

Linear Program to Compute Restricted Strategy

Restricted strategy : $\omega : V \Rightarrow V^k$

$\omega_{x,u}$: quantity of detectives on u when suspect is on x.

$f_{x,x',u,u'}$: quantity of detectives that go from u to u' when suspect goes from x to x'.

$$(1)\ \text{Minimize} \sum_{v \in V} \omega_{x_0,v}$$

Minimize number of detectives.

Linear Program

Restricted strategy : $\omega : V \Rightarrow V^k$

$\omega_{x,u}$: quantity of detectives on u when suspect is on x.

$f_{x,x',u,u'}$: quantity of detectives that go from u to u' when suspect goes from x to x'.

$$(2) \sum_{v \in N_d[x]} \omega_{x,v} \geq 1 \qquad \forall x \in V$$

Guarantees always at least 1 detective within distance d of suspect.

Linear Program

Restricted strategy : $\omega : V \Rightarrow V^k$

$\omega_{x,u}$: quantity of detectives on u when suspect is on x.

$f_{x,x',u,u'}$: quantity of detectives that go from u to u' when suspect goes from x to x'.

$$(3) \quad \sum_{u' \in N[u]} f_{x,x',u,u'} = \omega_{x,u} \qquad \forall u \in V,\ x' \in N_s[x]$$

$$(4) \quad \sum_{u' \in N[u]} f_{x,x',u',u} = \omega_{x',u} \qquad \forall u \in V,\ x' \in N_s[x]$$

Guarantees validity of moves of detectives when suspect moves.

Linear Program

Restricted strategy : $\omega : V \Rightarrow V^k$

$\omega_{x,u}$: quantity of detectives on u when suspect is on x.

$f_{x,x',u,u'}$: quantity of detectives that go from u to u' when suspect goes from x to x'.

$$(3) \sum_{u' \in N[u]} f_{x,x',u,u'} = \omega_{x,u} \qquad \forall u \in V,\ x' \in N_s[x]$$

$$(4) \sum_{u' \in N[u]} f_{x,x',u',u} = \omega_{x',u} \qquad \forall u \in V,\ x' \in N_s[x]$$

Guarantees validity of moves of detectives when suspect moves.

$O(n^4)$ real variables and constraints.

Main Result : *gn* in Trees

Theorem

$\forall s > 1$, $d \geq 0$ and all trees T, $gn_{s,d}(T)$ and a corresponding strategy can be calculated in polynomial time.

Idea of proof : Linear Program can compute opt. frac. restr. strategy in polynomial time.

Run LP. From previous theorem, strategy is opt. frac.

Can transform opt. frac. into opt. int. in polynomial time.

Grids

Theorem

$\exists \beta > 0$, s.t. $\forall s > 1$, $d \geq 0$, $\Omega(n^{1+\beta}) \leq gn_{s,d}(G_{n \times n})$.

Idea of proof : Lower bound holds for fractional version.

Grids

Theorem

$\exists \beta > 0$, s.t. $\forall s > 1$, $d \geq 0$, $\Omega(n^{1+\beta}) \leq gn_{s,d}(G_{n \times n})$.

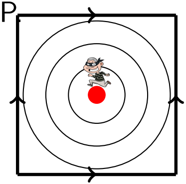
Idea of proof : Lower bound holds for fractional version.

Torus and grid have same order of number of detectives.

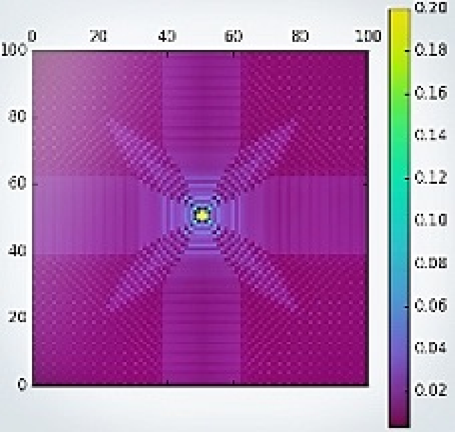
Theorem

$\exists \alpha \geq log(3/2) \approx 0.58$, s.t. $\forall s > 1$, $d \geq 0$,
$fgn_{s,d}(G_{n \times n}) \leq O(n^{2-\alpha})$.

Idea of proof : Density function $\omega^*(v) = \frac{c}{(dist(v,v_0)+1)^{log 3/2}}$ for a constant $c > 0$ satisfies LP.



Distribution of Detectives in the Torus for an optimal symmetrical suspect-positional strategy when $n = 100$, $m = 100$, $s = 2$ and $d = 1$



Further Work

- Determine $gn_{s,d}(G_{n \times n})$.
- Approximate $gn_{s,d}(G)$ in polynomial time in certain classes of graphs ?
- Fractional approach applied to other combinatorial games.

Thanks !